RAiO

RA8875

Character/Graphic

TFT LCD Controller

Application Note

Version 1.5

June, 28, 2013

RAiO Technology Inc.

Update History		
Version	**Date**	**Description**
1.0	May, 03, 2011	First Release
1.1	May,09,2011	Schematic 2 update ,initial code update
1.2	June,22,2011	Schematic 2 update
1.3	July,11,2011	Add chapter 6.appendix
1.4	Nov,30,2011	All schematic update, add serial IF,serial flash program circuit.
1.5	June,28,2013	Schematic 3 update ,initial code update

Chapter Contents Page

1. Application Circuit

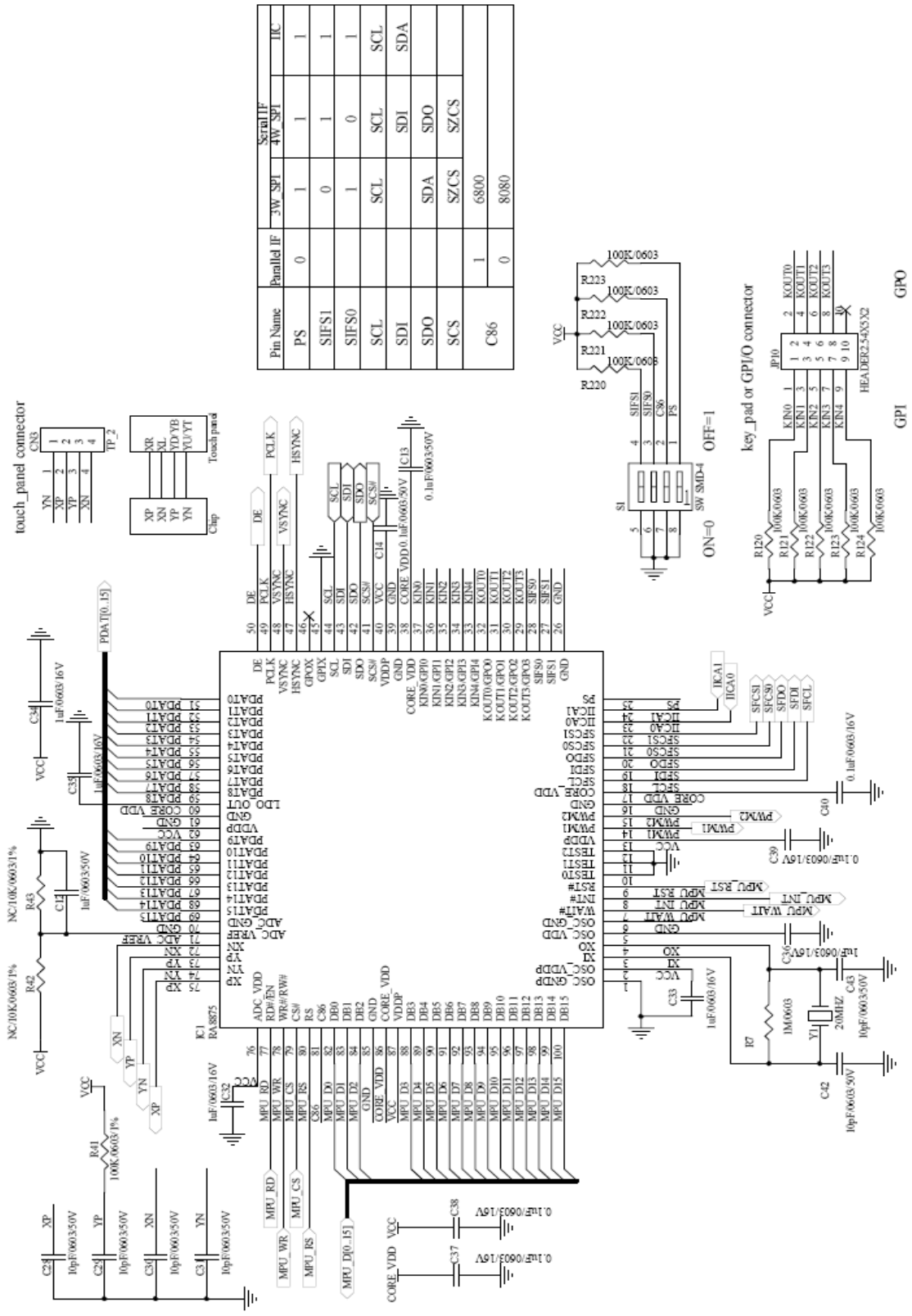


Figure 1-1

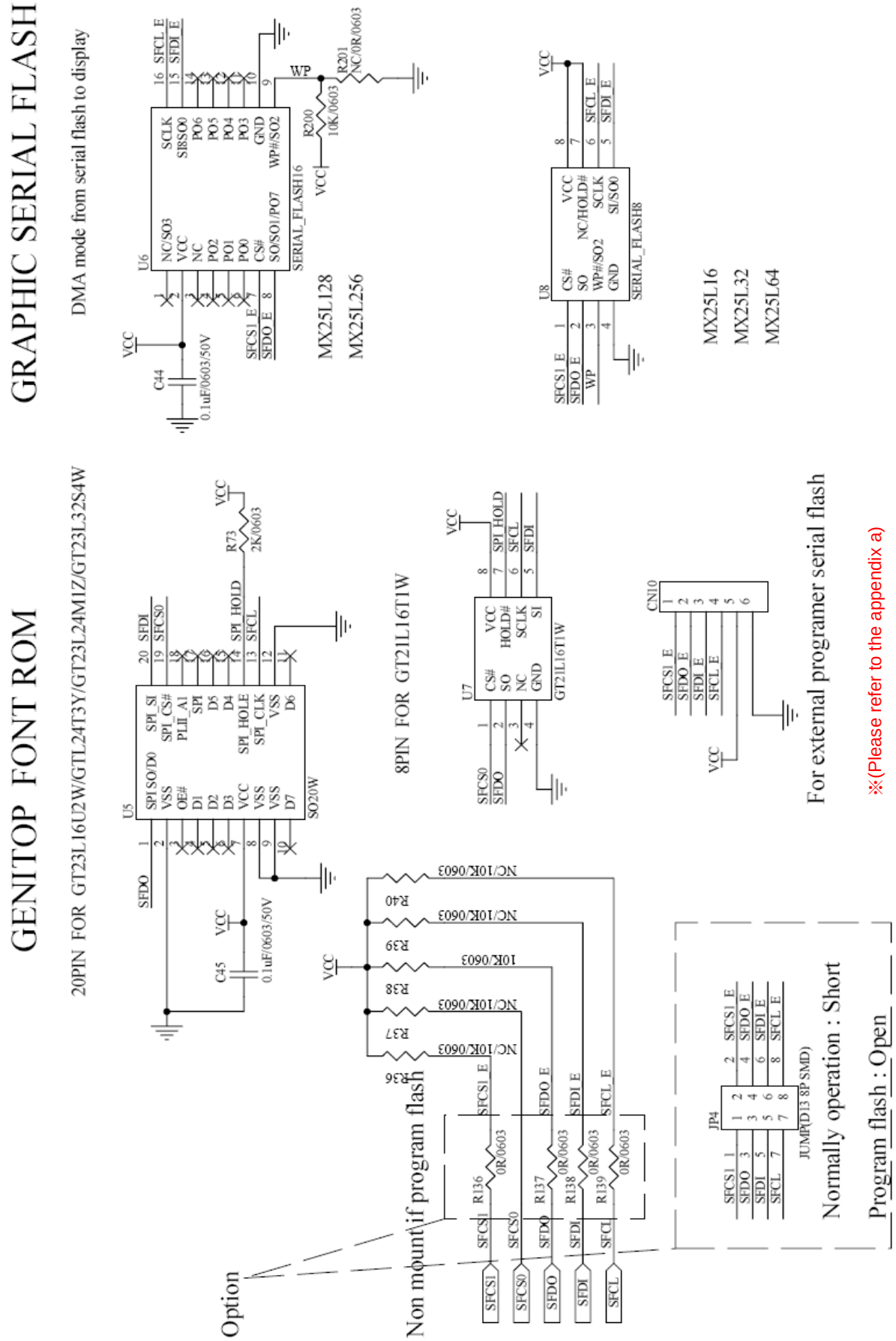

GRAPHIC SERIAL FLASH
DMA mode from serial flash to display
U6
NC/SO3
VCC
NC
PO2
PO1
PO0
CS#
SO/SO1/PO7
SCLK
SI/SO0
PO6
PO5
PO4
PO3
GND
WP#/SO2
SERIAL_FLASH16
WP
R200
10K/0603
R201
NC/0R/0603
C44
0.1uF/0603/50V
VCC
SFCL_E
SFDI_E
SFCS1_E
SFDO_E
MX25L128
MX25L256
U8
CS#
SO
WP#/SO2
GND
VCC
NC/HOLD#
SCLK
SI/SO0
SERIAL_FLASH8
MX25L16
MX25L32
MX25L64
GENITOP FONT ROM
20PIN FOR GT23L16U2W/GTL24T3Y/GT23L24M1Z/GT23L32S4W
U5
SPI SO/D0
VSS
OE#
D1
D2
D3
VCC
VSS
VSS
D7
SPI_SI
SPI_CS#
PLII_A1
SPI
D5
D4
SPI_HOLE
SPI_CLK
VSS
D6
SO20W
SFDO
SFDI
SFCS0
SPI_HOLD
SFCL
R73
2K/0603
C45
0.1uF/0603/50V
8PIN FOR GT21L16T1W
U7
CS#
SO
NC
GND
VCC
HOLD#
SCLK
SI
GT21L16T1W
SPI_HOLD
SFCL
SFDI
SFCS0
SFDO
CN10
For external programer serial flash
※(Please refer to the appendix a)
R40
R39
R38
R37
R36
NC/10K/0603
10K/0603
Option
Non mount if program flash
R136
R137
R138
R139
0R/0603
JP4
JUMP(D13 8P SMD)
Normally operation : Short
Program flash : Open

Figure 1-2

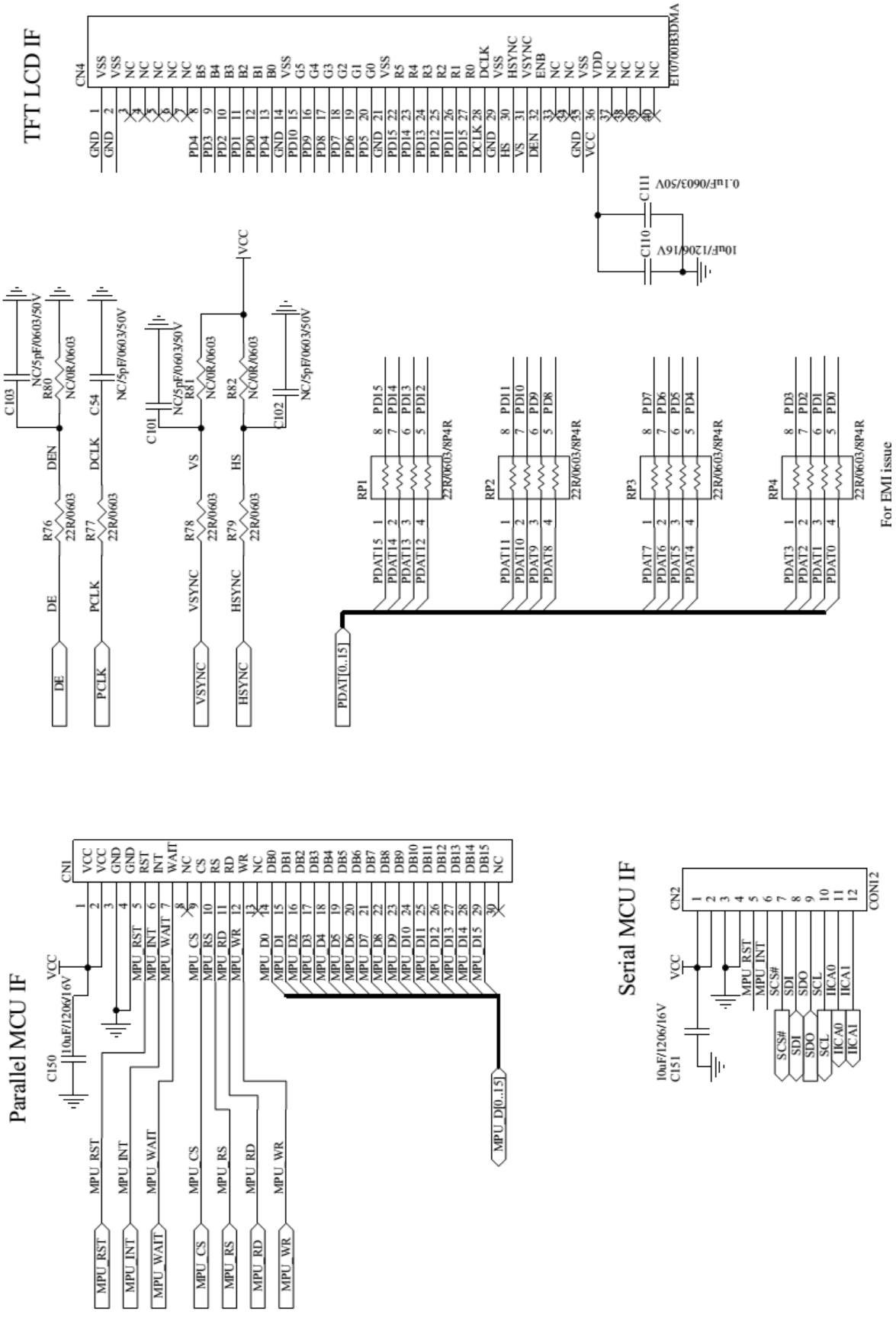

TFT LCD IF
CN4
ET070080B3DMA
C111
0.1uF/0603/50V
C110
10uF/1206/16V
C103
NC/5pF/0603/50V
R80
NC/0R/0603
C54
NC/5pF/0603/50V
C101
NC/5pF/0603/50V
R81
NC/0R/0603
R82
NC/0R/0603
C102
NC/5pF/0603/50V
VCC
R76
22R/0603
R77
22R/0603
R78
22R/0603
R79
22R/0603
DE
PCLK
VSYNC
HSYNC
RP1
RP2
RP3
RP4
22R/0603/8P4R
PDAT[0..15]
For EMI issue
Parallel MCU IF
CN1
C150
10uF/1206/16V
MPU_RST
MPU_INT
MPU_WAIT
MPU_CS
MPU_RS
MPU_RD
MPU_WR
MPU_D[0..15]
Serial MCU IF
CN2
CON12
C151
10uF/1206/16V
SCS#
SDI
SDO
SCL
IICA0
IICA1

Figure 1-3

2. Initial Code

```
void RA8875_PLL_ini(void)
{   //Base on 20MHz crystal
  #ifdef P320x240    //system clock = 20*(10+1)/(2^2) = 55MHz
    LCD_CmdWrite(0x88);
    LCD_DataWrite(0x0a);
    Delay1ms(1);
    LCD_CmdWrite(0x89);
    LCD_DataWrite(0x02);
    Delay1ms(1);
  #endif


  #ifdef P480x272     //system clock = 20*(10+1)/(2^2) = 55MHz
    LCD_CmdWrite(0x88);
    LCD_DataWrite(0x0a);
    Delay1ms(1);
    LCD_CmdWrite(0x89);
    LCD_DataWrite(0x02);
    Delay1ms(1);
  #endif

  #ifdef P640x480     //system clock = 20*(11+1)/(2^2) = 60MHz
    LCD_CmdWrite(0x88);
    LCD_DataWrite(0x0b);
    Delay1ms(1);
    LCD_CmdWrite(0x89);
    LCD_DataWrite(0x02);
    Delay1ms(1);
  #endif

  #ifdef P800x480     //system clock = 20*(11+1)/(2^2) = 60MHz
    LCD_CmdWrite(0x88);
    LCD_DataWrite(0x0b);
    Delay1ms(1);
    LCD_CmdWrite(0x89);
    LCD_DataWrite(0x02);
    Delay1ms(1);
```

```
#endif
}

//-----------------------------------------//
void LCD_Initial(void)
{
    RA8875_PLL_ini();
    LCD_CmdWrite(0x10);   //SYSR     bit[4:3]=00 256 color   bit[2:1]=   00 8bit MPU interface
    LCD_DataWrite(0x0c);     // if 8bit MCU interface     and 65k color display
    //LCD_DataWrite(0x0F); // if 16bit MCU interface     and 65k color display
#ifdef P320x240
//==============   Display Window320x240 ==================
    LCD_CmdWrite(0x04);        //set PCLK invers
    LCD_DataWrite(0x03);       //PCLK = system clock/(2^3) = 55/8 = 6.875MHz
    Delay1ms(1);

     //Horizontal set
    LCD_CmdWrite(0x14); //HDWR//Horizontal Display Width Setting Bit[6:0]
    LCD_DataWrite(0x27);//Horizontal display width(pixels) = (HDWR + 1)*8
    LCD_CmdWrite(0x15); //Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
    LCD_DataWrite(0x00);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
    LCD_CmdWrite(0x16); //HNDR//Horizontal Non-Display Period Bit[4:0]
    LCD_DataWrite(0x05);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
    LCD_CmdWrite(0x17); //HSTR//HSYNC Start Position[4:0]
    LCD_DataWrite(0x04);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
    LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
    LCD_DataWrite(0x03);//HSYNC Width [4:0]     HSYNC Pulse width(PCLK) = (HPWR + 1)*8
     //Vertical set
    LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0]
    LCD_DataWrite(0xef);//Vertical pixels = VDHR + 1
    LCD_CmdWrite(0x1A); //VDHR1 //Vertical Display Height Bit [8]
    LCD_DataWrite(0x00);//Vertical pixels = VDHR + 1
    LCD_CmdWrite(0x1B); //VNDR0 //Vertical Non-Display Period Bit [7:0]
    LCD_DataWrite(0x05);//Vertical Non-Display area = (VNDR + 1)
    LCD_CmdWrite(0x1C); //VNDR1 //Vertical Non-Display Period Bit [8]
    LCD_DataWrite(0x00);//Vertical Non-Display area = (VNDR + 1)
    LCD_CmdWrite(0x1D); //VSTR0 //VSYNC Start Position[7:0]
    LCD_DataWrite(0x0e);//VSYNC Start Position(PCLK) = (VSTR + 1)
    LCD_CmdWrite(0x1E); //VSTR1 //VSYNC Start Position[8]
    LCD_DataWrite(0x00);//VSYNC Start Position(PCLK) = (VSTR + 1)
```

```
    LCD_CmdWrite(0x1F); //VPWR //VSYNC Polarity ,VSYNC Pulse Width[6:0]
    LCD_DataWrite(0x02);//VSYNC Pulse Width(PCLK) = (VPWR + 1)

  //Active window   set
  //setting active window X
    LCD_CmdWrite(0x30); //Horizontal Start Point 0 of Active Window (HSAW0)
    LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [7:0]
    LCD_CmdWrite(0x31); //Horizontal Start Point 1 of Active Window (HSAW1)
    LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [9:8]
    LCD_CmdWrite(0x34); //Horizontal End Point 0 of Active Window (HEAW0)
    LCD_DataWrite(0x3F); //Horizontal End Point of Active Window [7:0]
    LCD_CmdWrite(0x35); //Horizontal End Point 1 of Active Window (HEAW1)
    LCD_DataWrite(0x01); //Horizontal End Point of Active Window [9:8]

   //setting active window Y
    LCD_CmdWrite(0x32); //Vertical Start Point 0 of Active Window (VSAW0)
    LCD_DataWrite(0x00); //Vertical Start Point of Active Window [7:0]
    LCD_CmdWrite(0x33); //Vertical Start Point 1 of Active Window (VSAW1)
    LCD_DataWrite(0x00); //Vertical Start Point of Active Window [8]
    LCD_CmdWrite(0x36); //Vertical End Point of Active Window 0 (VEAW0)
    LCD_DataWrite(0xef); //Vertical End Point of Active Window [7:0]
    LCD_CmdWrite(0x37); //Vertical End Point of Active Window 1 (VEAW1)
    LCD_DataWrite(0x00); //Vertical End Point of Active Window [8]

#endif

  #ifdef P480x272
    LCD_CmdWrite(0x04);       //set PCLK invers
    LCD_DataWrite(0x82);      //PCLK = system clock/(2^2) = 55/4 = 13.75 MHz
    Delay1ms(1);

    //Horizontal set
    LCD_CmdWrite(0x14); //HDWR//Horizontal Display Width Setting Bit[6:0]
    LCD_DataWrite(0x3B);//Horizontal display width(pixels) = (HDWR + 1)*8
    LCD_CmdWrite(0x15); //Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
    LCD_DataWrite(0x00);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
    LCD_CmdWrite(0x16); //HNDR//Horizontal Non-Display Period Bit[4:0]
    LCD_DataWrite(0x01);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
    LCD_CmdWrite(0x17); //HSTR//HSYNC Start Position[4:0]
    LCD_DataWrite(0x00);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
```

```
LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
LCD_DataWrite(0x05);//HSYNC Width [4:0]     HSYNC Pulse width(PCLK) = (HPWR + 1)*8

//Vertical set
LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0]
LCD_DataWrite(0x0f); //Vertical pixels = VDHR + 1
LCD_CmdWrite(0x1a); //VDHR1 //Vertical Display Height Bit [8]
LCD_DataWrite(0x01); //Vertical pixels = VDHR + 1
LCD_CmdWrite(0x1b);    //VNDR0 //Vertical Non-Display Period Bit [7:0]
LCD_DataWrite(0x02);    //VSYNC Start Position(PCLK) = (VSTR + 1)
LCD_CmdWrite(0x1c);    //VNDR1 //Vertical Non-Display Period Bit [8]
LCD_DataWrite(0x00); //Vertical Non-Display area = (VNDR + 1)
LCD_CmdWrite(0x1d); //VSTR0 //VSYNC Start Position[7:0]
LCD_DataWrite(0x07);//VSYNC Start Position(PCLK) = (VSTR + 1)
LCD_CmdWrite(0x1e); //VSTR1 //VSYNC Start Position[8]
LCD_DataWrite(0x00);//VSYNC Start Position(PCLK) = (VSTR + 1)
LCD_CmdWrite(0x1f);       //VPWR //VSYNC Polarity ,VSYNC Pulse Width[6:0]
LCD_DataWrite(0x09);      //VSYNC Pulse Width(PCLK) = (VPWR + 1)

//Active window    set
//setting active window X
LCD_CmdWrite(0x30); //Horizontal Start Point 0 of Active Window (HSAW0)
LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [7:0]
LCD_CmdWrite(0x31); //Horizontal Start Point 1 of Active Window (HSAW1)
LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [9:8]
LCD_CmdWrite(0x34); //Horizontal End Point 0 of Active Window (HEAW0)
LCD_DataWrite(0xDF); //Horizontal End Point of Active Window [7:0]
LCD_CmdWrite(0x35); //Horizontal End Point 1 of Active Window (HEAW1)
LCD_DataWrite(0x01); //Horizontal End Point of Active Window [9:8]

//setting active window Y
LCD_CmdWrite(0x32); //Vertical Start Point 0 of Active Window (VSAW0)
LCD_DataWrite(0x00); //Vertical Start Point of Active Window [7:0]
LCD_CmdWrite(0x33); //Vertical Start Point 1 of Active Window (VSAW1)
LCD_DataWrite(0x00); //Vertical Start Point of Active Window [8]
LCD_CmdWrite(0x36); //Vertical End Point of Active Window 0 (VEAW0)
LCD_DataWrite(0x0F); //Vertical End Point of Active Window [7:0]
LCD_CmdWrite(0x37); //Vertical End Point of Active Window 1 (VEAW1)
LCD_DataWrite(0x01); //Vertical End Point of Active Window [8]
```

```
#endif

#ifdef P640x480
//==============  Display Window640x480 ==================
      LCD_CmdWrite(0x04);       //PCLK inverse
      LCD_DataWrite(0x01);      //PCLK = system clock/2 = 60/2 = 30 MHz
      Delay1ms(1);
      //Horizontal set
      LCD_CmdWrite(0x14);//HDWR//Horizontal Display Width Setting Bit[6:0]
      LCD_DataWrite(0x4F);//Horizontal display width(pixels) = (HDWR + 1)*8
      LCD_CmdWrite(0x15); //Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
      LCD_DataWrite(0x05);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
      LCD_CmdWrite(0x16);//HNDR//Horizontal Non-Display Period Bit[4:0]
      LCD_DataWrite(0x0f);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
      LCD_CmdWrite(0x17);//HSTR//HSYNC Start Position[4:0]
      LCD_DataWrite(0x01);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
      LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
      LCD_DataWrite(0x00);//HSYNC Width [4:0]     HSYNC Pulse width(PCLK) = (HPWR + 1)*8
      //Vertical set
      LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0]
      LCD_DataWrite(0xdf);//Vertical pixels = VDHR + 1
      LCD_CmdWrite(0x1A);//VDHR1 //Vertical Display Height Bit [8]
      LCD_DataWrite(0x01);//Vertical pixels = VDHR + 1
      LCD_CmdWrite(0x1B);//VNDR0 //Vertical Non-Display Period Bit [7:0]
      LCD_DataWrite(0x0A);//Vertical Non-Display area = (VNDR + 1)
      LCD_CmdWrite(0x1C);//VNDR1 //Vertical Non-Display Period Bit [8]
      LCD_DataWrite(0x00);//Vertical Non-Display area = (VNDR + 1)
      LCD_CmdWrite(0x1D);//VSTR0 //VSYNC Start Position[7:0]
      LCD_DataWrite(0x0E);//VSYNC Start Position(PCLK) = (VSTR + 1)
      LCD_CmdWrite(0x1E);//VSTR1 //VSYNC Start Position[8]
      LCD_DataWrite(0x00);//VSYNC Start Position(PCLK) = (VSTR + 1)
      LCD_CmdWrite(0x1F);//VPWR //VSYNC Polarity ,VSYNC Pulse Width[6:0]
      LCD_DataWrite(0x01);//VSYNC Pulse Width(PCLK) = (VPWR + 1)

     //Active window    set
     //setting active window X
      LCD_CmdWrite(0x30); //Horizontal Start Point 0 of Active Window (HSAW0)
      LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [7:0]
      LCD_CmdWrite(0x31); //Horizontal Start Point 1 of Active Window (HSAW1)
      LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [9:8]
```

```
    LCD_CmdWrite(0x34); //Horizontal End Point 0 of Active Window (HEAW0)
    LCD_DataWrite(0x7f); //Horizontal End Point of Active Window [7:0]
    LCD_CmdWrite(0x35); //Horizontal End Point 1 of Active Window (HEAW1)
    LCD_DataWrite(0x02); //Horizontal End Point of Active Window [9:8]

  //setting active window Y
    LCD_CmdWrite(0x32); //Vertical Start Point 0 of Active Window (VSAW0)
    LCD_DataWrite(0x00); //Vertical Start Point of Active Window [7:0]
    LCD_CmdWrite(0x33); //Vertical Start Point 1 of Active Window (VSAW1)
    LCD_DataWrite(0x00); //Vertical Start Point of Active Window [8]
    LCD_CmdWrite(0x36); //Vertical End Point of Active Window 0 (VEAW0)
    LCD_DataWrite(0xdf); //Vertical End Point of Active Window [7:0]
    LCD_CmdWrite(0x37); //Vertical End Point of Active Window 1 (VEAW1)
    LCD_DataWrite(0x01); //Vertical End Point of Active Window [8]

#endif


#ifdef P800x480

//AT070TN92    setting
//==============   Display Window800x480 ==================
  /*
    LCD_CmdWrite(0x04);        //PCLK inverse
    LCD_DataWrite(0x81);       //PCLK = system clock/2 = 60/2 = 30 MHz
    Delay1ms(1);
    //Horizontal set
    LCD_CmdWrite(0x14); //HDWR//Horizontal Display Width Setting Bit[6:0]
    LCD_DataWrite(0x63);//Horizontal display width(pixels) = (HDWR + 1)*8
    LCD_CmdWrite(0x15); //Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
    LCD_DataWrite(0x03);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
    LCD_CmdWrite(0x16); //HNDR//Horizontal Non-Display Period Bit[4:0]
    LCD_DataWrite(0x03);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
    LCD_CmdWrite(0x17); //HSTR//HSYNC Start Position[4:0]
    LCD_DataWrite(0x02);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
    LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
    LCD_DataWrite(0x00);//HSYNC Width [4:0]     HSYNC Pulse width(PCLK) = (HPWR + 1)*8
    //Vertical set
    LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0]
    LCD_DataWrite(0xdf);//Vertical pixels = VDHR + 1
```

```
    LCD_CmdWrite(0x1a); //VDHR1 //Vertical Display Height Bit [8]
    LCD_DataWrite(0x01);//Vertical pixels = VDHR + 1
    LCD_CmdWrite(0x1b); //VNDR0 //Vertical Non-Display Period Bit [7:0]
    LCD_DataWrite(0x14);//Vertical Non-Display area = (VNDR + 1)
    LCD_CmdWrite(0x1c); //VNDR1 //Vertical Non-Display Period Bit [8]
    LCD_DataWrite(0x00);//Vertical Non-Display area = (VNDR + 1)
    LCD_CmdWrite(0x1d); //VSTR0 //VSYNC Start Position[7:0]
    LCD_DataWrite(0x06);//VSYNC Start Position(PCLK) = (VSTR + 1)
    LCD_CmdWrite(0x1e); //VSTR1 //VSYNC Start Position[8]
    LCD_DataWrite(0x00);//VSYNC Start Position(PCLK) = (VSTR + 1)
    LCD_CmdWrite(0x1f); //VPWR //VSYNC Polarity ,VSYNC Pulse Width[6:0]
    LCD_DataWrite(0x01);//VSYNC Pulse Width(PCLK) = (VPWR + 1)
  //LCD_CmdWrite(0xf2);
  //LCD_DataWrite(0x01);
     */
//HSD050IDW1 setting
//==============   Display Window800x480 ==================
  LCD_CmdWrite(0x04);    //PCLK inverse
  LCD_DataWrite(0x81);
  Delay1ms(1);
  //Horizontal set
  LCD_CmdWrite(0x14); //HDWR//Horizontal Display Width Setting Bit[6:0]
  LCD_DataWrite(0x63);//Horizontal display width(pixels) = (HDWR + 1)*8
  LCD_CmdWrite(0x15);//Horizontal Non-Display Period Fine Tuning Option Register (HNDFTR)
  LCD_DataWrite(0x00);//Horizontal Non-Display Period Fine Tuning(HNDFT) [3:0]
  LCD_CmdWrite(0x16); //HNDR//Horizontal Non-Display Period Bit[4:0]
  LCD_DataWrite(0x03);//Horizontal Non-Display Period (pixels) = (HNDR + 1)*8
  LCD_CmdWrite(0x17); //HSTR//HSYNC Start Position[4:0]
  LCD_DataWrite(0x03);//HSYNC Start Position(PCLK) = (HSTR + 1)*8
  LCD_CmdWrite(0x18); //HPWR//HSYNC Polarity ,The period width of HSYNC.
  LCD_DataWrite(0x0B);//HSYNC Width [4:0]     HSYNC Pulse width(PCLK) = (HPWR + 1)*8
  //Vertical set
  LCD_CmdWrite(0x19); //VDHR0 //Vertical Display Height Bit [7:0]
  LCD_DataWrite(0xdf);//Vertical pixels = VDHR + 1
  LCD_CmdWrite(0x1a); //VDHR1 //Vertical Display Height Bit [8]
  LCD_DataWrite(0x01);//Vertical pixels = VDHR + 1
  LCD_CmdWrite(0x1b); //VNDR0 //Vertical Non-Display Period Bit [7:0]
  LCD_DataWrite(0x20);//Vertical Non-Display area = (VNDR + 1)
  LCD_CmdWrite(0x1c); //VNDR1 //Vertical Non-Display Period Bit [8]
  LCD_DataWrite(0x00);//Vertical Non-Display area = (VNDR + 1)
```

```
 LCD_CmdWrite(0x1d); //VSTR0 //VSYNC Start Position[7:0]
 LCD_DataWrite(0x16);//VSYNC Start Position(PCLK) = (VSTR + 1)
 LCD_CmdWrite(0x1e); //VSTR1 //VSYNC Start Position[8]
 LCD_DataWrite(0x00);//VSYNC Start Position(PCLK) = (VSTR + 1)
 LCD_CmdWrite(0x1f); //VPWR //VSYNC Polarity ,VSYNC Pulse Width[6:0]
 LCD_DataWrite(0x01);//VSYNC Pulse Width(PCLK) = (VPWR + 1)

  //Active window   set
   //setting active window X
    LCD_CmdWrite(0x30); //Horizontal Start Point 0 of Active Window (HSAW0)
    LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [7:0]
    LCD_CmdWrite(0x31); //Horizontal Start Point 1 of Active Window (HSAW1)
    LCD_DataWrite(0x00); //Horizontal Start Point of Active Window [9:8]
    LCD_CmdWrite(0x34); //Horizontal End Point 0 of Active Window (HEAW0)
    LCD_DataWrite(0x1F); //Horizontal End Point of Active Window [7:0]
    LCD_CmdWrite(0x35); //Horizontal End Point 1 of Active Window (HEAW1)
    LCD_DataWrite(0x03); //Horizontal End Point of Active Window [9:8]

   //setting active window Y
    LCD_CmdWrite(0x32); //Vertical Start Point 0 of Active Window (VSAW0)
    LCD_DataWrite(0x00); //Vertical Start Point of Active Window [7:0]
    LCD_CmdWrite(0x33); //Vertical Start Point 1 of Active Window (VSAW1)
    LCD_DataWrite(0x00); //Vertical Start Point of Active Window [8]
    LCD_CmdWrite(0x36); //Vertical End Point of Active Window 0 (VEAW0)
    LCD_DataWrite(0xdf); //Vertical End Point of Active Window [7:0]
    LCD_CmdWrite(0x37); //Vertical End Point of Active Window 1 (VEAW1)
    LCD_DataWrite(0x01); //Vertical End Point of Active Window [8]
#endif
}
```

3. Display on Sequence

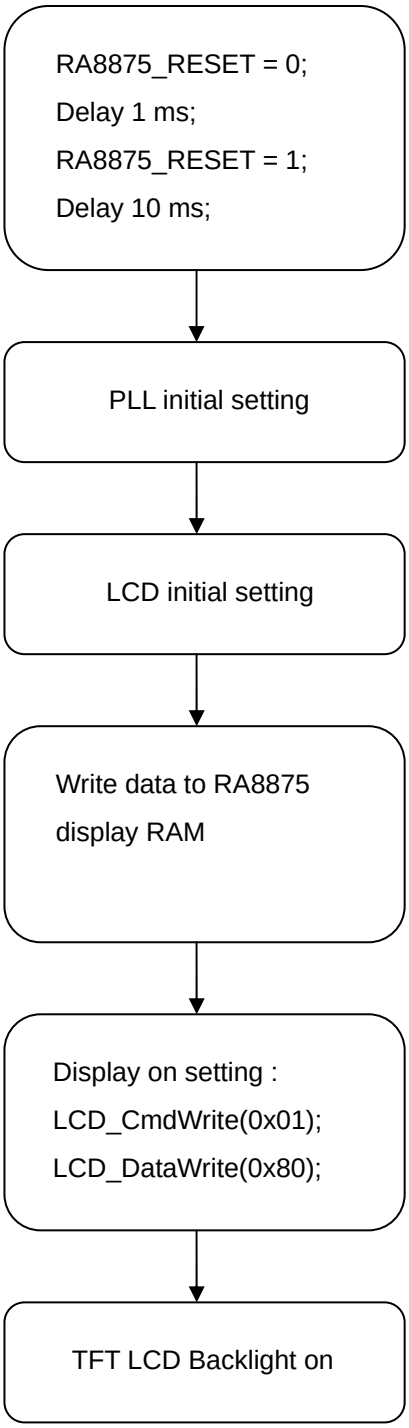


Figure 3-1

4. Sleep Mode Sequence

Enter Sleep :

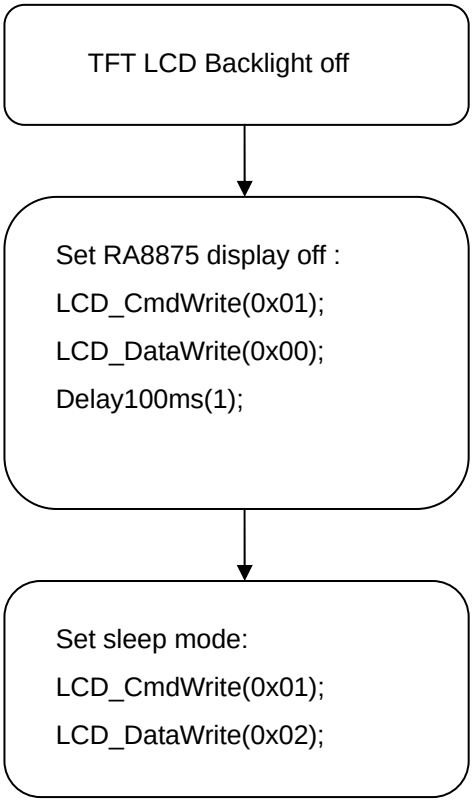


Figure 4-1

*Attention:

We suggest to confirm your TFT panel could be disable when the RA8875 enter sleep, to avoid liquid crystal polarization.RAiO shall not be held liable for any damages about TFT panel ,when customers use sleep mode incorrect !

Exit Sleep :

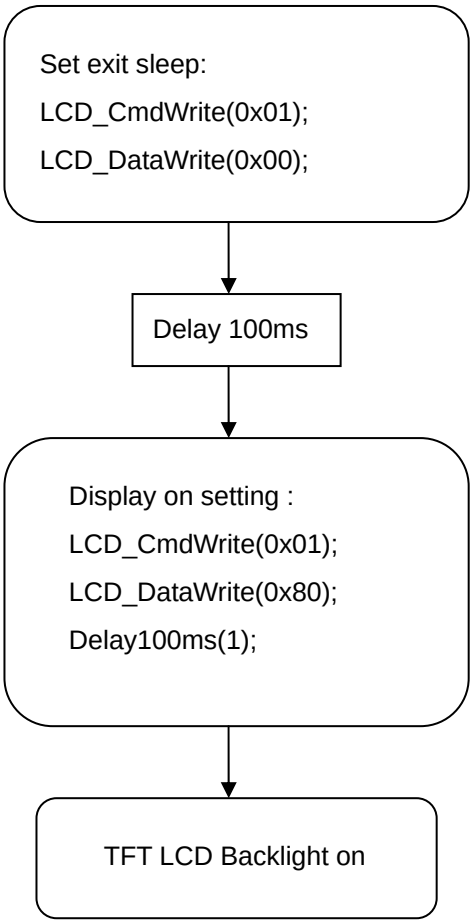


Figure 4-2

5. Display RAM pure data write example

Include RA8875_subrotine.c first , then refer the code below.

If MCU 8 bit interface :

```
Graphic_Mode( );    //set to graphic mode
XY_Coordinate(0,0); //set write cursor position
LCD_CmdWrite(0x02);//set CMD [02h] before data write
 for (i=384000;i>0;i--)
 { LCD_DataWrite(0xf8); // write color red data
   LCD_DataWrite(0x00);
 }
```

If MCU 16 bit interface :

```
Graphic_Mode( );    //set to graphic mode
XY_Coordinate(0,0); //set write cursor position
LCD_CmdWrite(0x02);// //set CMD [02h] before data write
 for (i=384000;i>0;i--)
 { LCD_DataWrite(0xf800); }// write color red data
```

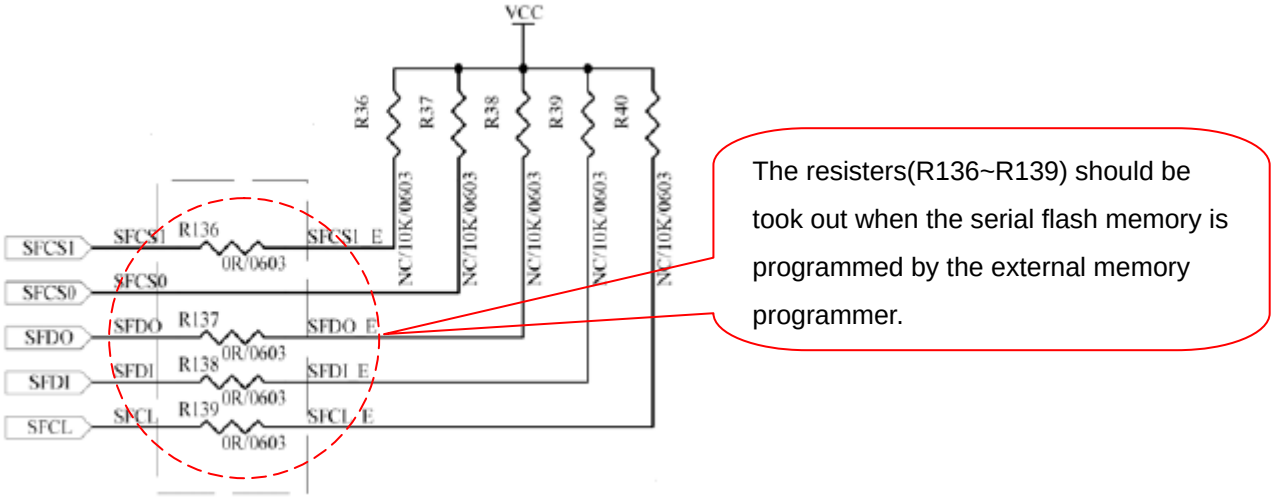
You could download some example code and RA8875 subrotine on the website :

www.raio.com.tw

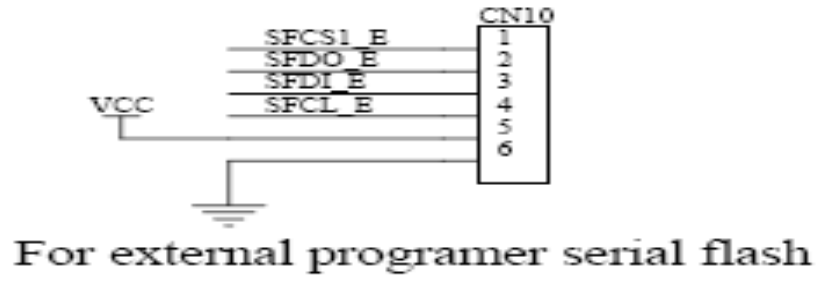
6. Appendix

a. How to program the Serial Flash Memory by the external programmer?

If the advance circuit is already designed by the LCM factory as the following picture1 and picture2, just leading the related pins of the serial flash out to the connector and then the user will be able to program the picture data(*.bin) by the external memory programmer. Please refer to the following pinture3.

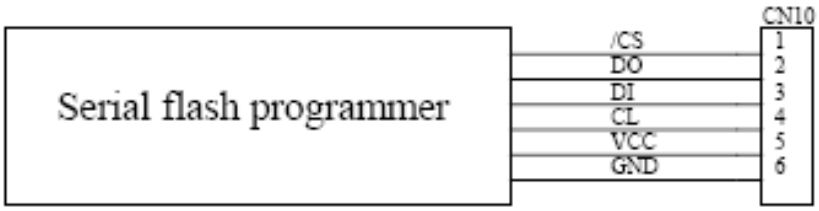


Picture 1



For external programer serial flash

Picture2



Picture3